Orca

A DeepSeek-native local terminal coding agent with long-context workflows, resumable history, and approval-aware automation.

About

Orca is a coding agent for the terminal that is built around DeepSeek. It focuses on long-context coding, multi-turn tool use, prefix-cache-friendly prompts, persistent goals, resumable history, and approval-aware automation so you can keep work moving inside one local workflow. The app is delivered as a Rust-native CLI and TUI, and its design is centered on readable, auditable coding runs rather than a hidden agent loop.

What is Orca

Orca is a DeepSeek-native local terminal coding agent for developers who want an

terminal UI, so the experience stays close to the code, the repo, and the tools you already use. Orca is designed for long-context coding, multi-turn tool use, persistent goals, resumable history, and approval-aware automation in one place.

The app is specifically tuned for DeepSeek rather than a generic model-agnostic shell. That matters because Orca emphasizes prefix-cache behavior, SSE streaming, and DeepSeek reasoning defaults. The site also shows version v0.2.36 and highlights support for a 1M context window, 128 max turns, and 16 tool surfaces.

Key features

Orca combines a terminal coding agent with a set of controls that make long sessions easier to manage.

- **1M context window**: the tool advertises a 1 million token window, with automatic compaction once the conversation crosses an 80% threshold.
- **DeepSeek-native behavior**: it uses DeepSeek reasoning defaults, SSE streaming, and tool-use semantics that are aligned with that model family.
- **Prefix-cache friendly prompts**: the prompt structure is append-only at the byte level, which helps preserve prefix cache stability across multi-turn work.
- **Persistent goals**: you can create a long-running objective with `/goal`, and the agent can continue after successful turns.
- **Resumable history**: local JSONL transcripts support list, search, `--resume`, `--fork`, archive actions, and optional zstd compression.
- **Approval-aware automation**: reads can run directly, while writes, shell commands, network actions, and agent actions follow your configured policy.
- **Skills and user input**: Markdown skills can be injected with `$skill` identifiers, and the TUI can ask structured `request_user_input` questions without ending the turn.
- **Verifier loop support**: you can gate runs with a command such as `--verifier "cargo test"`.

Orca also exposes a readable control surface with commands and tool actions such as `read_file`, `glob`, `edit`, `grep`, `bash`, `write_file`, `git_status`, `web_search`, `subagent`, and workflow operations like `update_plan` and goal management.

Orca is meant to be installed once and then launched from the terminal whenever you need a coding agent. The site shows an npm install path and a native curl installer path.

1. Install the CLI, either through npm or the native installer.
2. Run `orca` in your terminal.
3. Provide your DeepSeek API key during the first-run setup if it has not already been configured.
4. Type the task you want handled in the interactive TUI.
5. Use `/goal` for long tasks, `--resume` to continue a transcript, or `--fork` to branch a new run.
6. Add a verifier command when you want the run to end only after a real check passes.

Orca also supports reusable workflows from `.orca/workflows/`, which makes it useful when the same process needs to be repeated with the same shape and checks.

Who it is for

Orca is a good fit for developers who want a terminal coding agent that can stay inside a repository for longer than a single prompt.

- Engineers doing **fix-the-test** loops who want the transcript preserved for review.
- Teams working on **large refactors** that need a persistent goal instead of a one-shot reply.
- Developers doing **codebase archaeology** who need search, reading, and summaries without losing the reasoning path.
- People who want **release checks** that combine command execution, events, and notes into an auditable pass.
- Users who prefer a **local terminal coding agent** over a browser-based assistant.

Because Orca is DeepSeek-native, it is especially relevant for users who already plan to use DeepSeek API keys and want the agent workflow tuned around that model's prefix-cache behavior.

What to know before you use it

and agent actions can still require policy approval. That is useful for control, but it also means some tasks will pause for consent or sandbox rules.

A few practical caveats matter:

- Orca depends on a DeepSeek API key for first-time setup and model-backed use.
- The app is local and terminal-based; it is not presented as a hosted web app.
- The 1M context window helps with large sessions, but compaction still happens past the 80% threshold, so very long work may be summarized.
- Resumable history is stored in local JSONL transcripts under `~/.orca/sessions/`, so session management is a local responsibility.
- The agent can automate coding work, but it does **not** replace official checks for final release decisions, security validation, or production deployment approval.
- For exact behavior, policy details, and the latest commands, use the project's GitHub repository and documentation rather than relying on a summary.

If you want a DeepSeek-native coding agent that keeps context, goals, and history inside the terminal, Orca is built for that workflow. If you need a hosted collaboration surface or a fully managed service, this app is not that; if you need a local terminal coding agent with approval-aware automation, Orca is the better match.

FAQ

What is Orca?

How do I install Orca?

Is Orca free to use?

Which platforms does Orca support?

Does Orca require an API key?

What makes Orca different from other terminal coding agents?

How does Orca handle long coding sessions?

What makes Orca different from other coding agents?

What approval controls does Orca offer?

Can I pick up where I left off in a previous coding session?

How does Orca ensure safety for file writes and network actions?

Can Orca run commands automatically?

Can I resume previous coding sessions?

Does Orca support automated or reusable workflows?

What platforms does Orca support?

Who gets the most value from Orca?

Sponsored

Belay – Awa... Featured
Developer Tools

Stays awake while your agents work.

Partly Featured
Finance

Partly is a mobile app that lets friends

View ↗

View ↗

Visit website ↗

orca

Pricing	Free
Category	AI
Platform	Mac, Windows, Linux, API
Launched	Aug 2026
Updated	12 days ago
Verified	1 months ago

TAGS

ai # coding-agent # terminal # tui # automation # deepseek # rust # cli # open source

Dir Hub

A curated directory of small, well-made apps. Browse by category or search to find the right one in fewer clicks.

DISCOVER

Home

Pricing

Submit your app

COMPANY

LEGAL

O Orca

O Orca